**ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ**



**МОСКОВСКИЙ ПОЛИТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ**

**Факультет Информационных технологий**

***Кафедра Информатики и информационных технологий***

**направление подготовки**

**09.03.02 «Информационные системы и технологии»**

**ЛАБОРАТОРНАЯ РАБОТА № 14-15**

**Дисциплина:** Основы алгоритмизации и программирования

# Тема: "Алгоритм сортировки «Гномья»"

# Выполнил: студент группы 211-721

Дерендяев Дмитрий Сергеевич

(Фамилия И.О.)

**Дата, подпись** 4.12.2021  ***\_\_\_\_\_\_\_\_\_\_\_***

(Дата) (Подпись)

**Проверил: Новичков Иван Константинович *\_\_\_\_\_\_\_\_\_\_\_***

(Фамилия И.О., степень, звание) **(Оценка)**

**Дата, подпись** \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  ***\_\_\_\_\_\_\_\_\_\_\_***

(Дата) (Подпись)

**Замечания: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**Москва**

**2021**

# Лабораторная работа №14-15 "Алгоритм сортировки «Гномья»"

**Цель:** Получить практические навыки разработке алгоритмов и их программной реализации.

**Понятие алгоритма:**

**Гномья сортировка** (англ. *Gnome sort*) — алгоритм сортировки, похожий на сортировку вставками, но в отличие от последней перед вставкой на нужное место происходит серия обменов, как в сортировке пузырьком. Название происходит от предполагаемого поведения садовых гномов при сортировке линии садовых горшков.

Алгоритм концептуально простой, не требует вложенных циклов. Время работы O(n²). На практике алгоритм может работать так же быстро, как и сортировка вставками.

**Идея алгоритма:**

**Пусть имеется массив A размером N, тогда сортировка выбором сводится к следующему:**

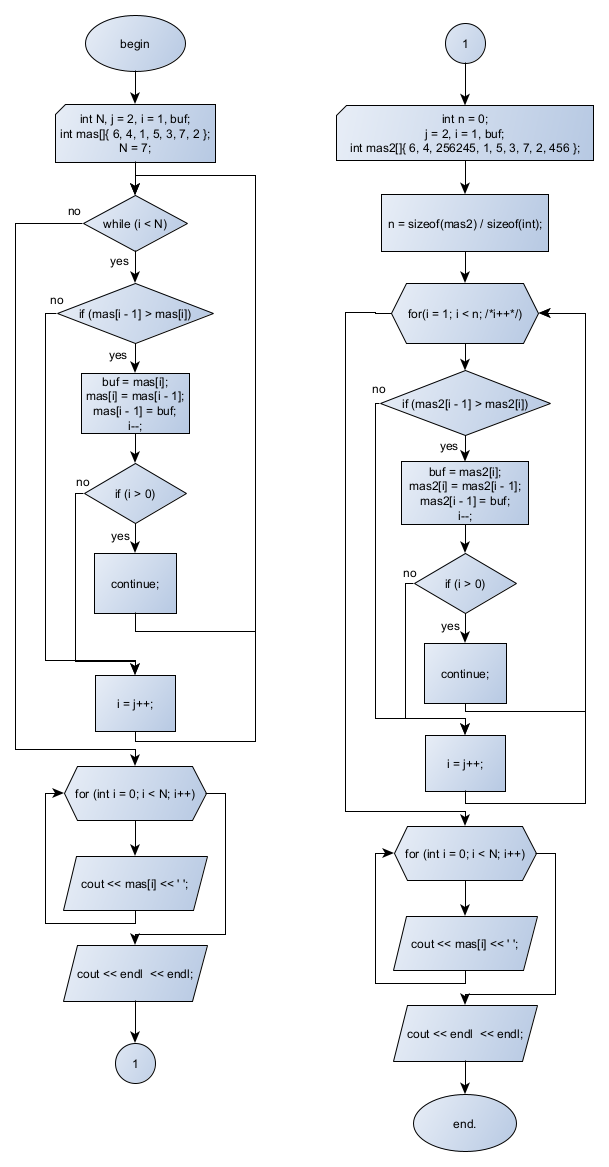
* Смотрим на текущий и предыдущий элемент массива:
* если они в правильном порядке, шагаем на один элемент вперед,
* иначе меняем их местами и шагаем на один элемент назад.
* Граничные условия:
* если нет предыдущего элемента, шагаем вперёд;
* если нет следующего элемента, стоп.

Это оптимизированная версия с использованием переменной j, чтобы разрешить прыжок вперёд туда, где он остановился до движения влево, избегая лишних итераций и сравнений.

**Задачи:**

Необходимо выполнить и оформить описание следующих пунктов:

1. Сформулировать идею алгоритма
2. Выполнить словесное представление алгоритма
3. Выполнить полнить представление алгоритма с помощью блок схем с использованием элемента модификации и без него.
4. Выполнить программную реализацию алгоритмов на языке С с использованием параметрического цикла и цикла с предусловием.

****

**Словесное описание алгоритма:**

1. **Сортировка начинается со второго и третьего элементов i=1, j=2;**
2. **Если i < N, то к пункту 3, иначе к пункту 9**
3. **если arr[i - 1] > arr[i], то к пункту 4, иначе к пункту 7**
4. **Меняем местами значения arr[i] и arr[i - 1]**
5. **Шагаем на один элемент назад i--**
6. **Если i > 0, то к пункту 2(используя оператор continue), иначе к пункту 7**
7. **i = j++**
8. **К пункту 2.**
9. **Конец алгоритма**

**Листинг программы:**

#include <iostream>//подключение необходимых библиотек

using namespace std;//определение пространаства имен

int main()

{

int N, j = 2, i = 1, buf;//подготовка необходимых переменных

//int \*mas = new int[7];

int mas[]{ 6, 4, 1, 5, 3, 7, 2 };//инициализация массива

N = 7;//размер массива вручную

while (i < N) {//начало цикла прохода по всему массиву

if (mas[i - 1] > mas[i]) //проверка условия, если предыдущий элемент больше текущего

{

buf = mas[i];//производим обмен значений переменнызх посредством буферной переменной

mas[i] = mas[i - 1];

mas[i - 1] = buf;

i--;//т.к. мы попали в условие, значит, у нас произошла перестановка и нам важно понять, меньше ли все предыдущие значения и правильный ли там порядок, делаем шаг назад

if (i > 0) //если i больше нуля, продолжаем итерацию цикла

continue;

}

i = j++;//приравниваем новый номер к заранее сохраненному(это позволяет алгоритму перепрыгивать на то место, откуда он начал уходить в анализ предыдущих элементов))

}

for (int i = 0; i < N; i++)// выводим массив на экран

{

cout << mas[i] << ' ';

}

cout << endl << endl;

//------------------------------------------------------

int n = 0;//подготовка необходимых переменных

j = 2, i = 1, buf;

//int \*mas = new int[7];

int mas2[]{ 6, 4, 256245, 1, 5, 3, 7, 2, 456 };//инициализация массива

n = sizeof(mas2) / sizeof(int);//определение размера массива

for(i = 1; i < n; /\*i++\*/)//начало цикла прохода по всему массиву

{

if (mas2[i - 1] > mas2[i])//проверка условия, если предыдущий элемент больше текущего

{

buf = mas2[i];//производим обмен значений переменнызх посредством буферной переменной

mas2[i] = mas2[i - 1];

mas2[i - 1] = buf;

i--;//т.к. мы попали в условие, значит, у нас произошла перестановка и нам важно понять, меньше ли все предыдущие значения и правильный ли там порядок, делаем шаг назад

if (i > 0)//если i больше нуля, продолжаем итерацию цикла

continue;

}

i = j++;//приравниваем новый номер к заранее сохраненному(это позволяет алгоритму перепрыгивать на то место, откуда он начал уходить в анализ предыдущих элементов))

}

for (int i = 0; i < n; i++)//вывод массива на экран

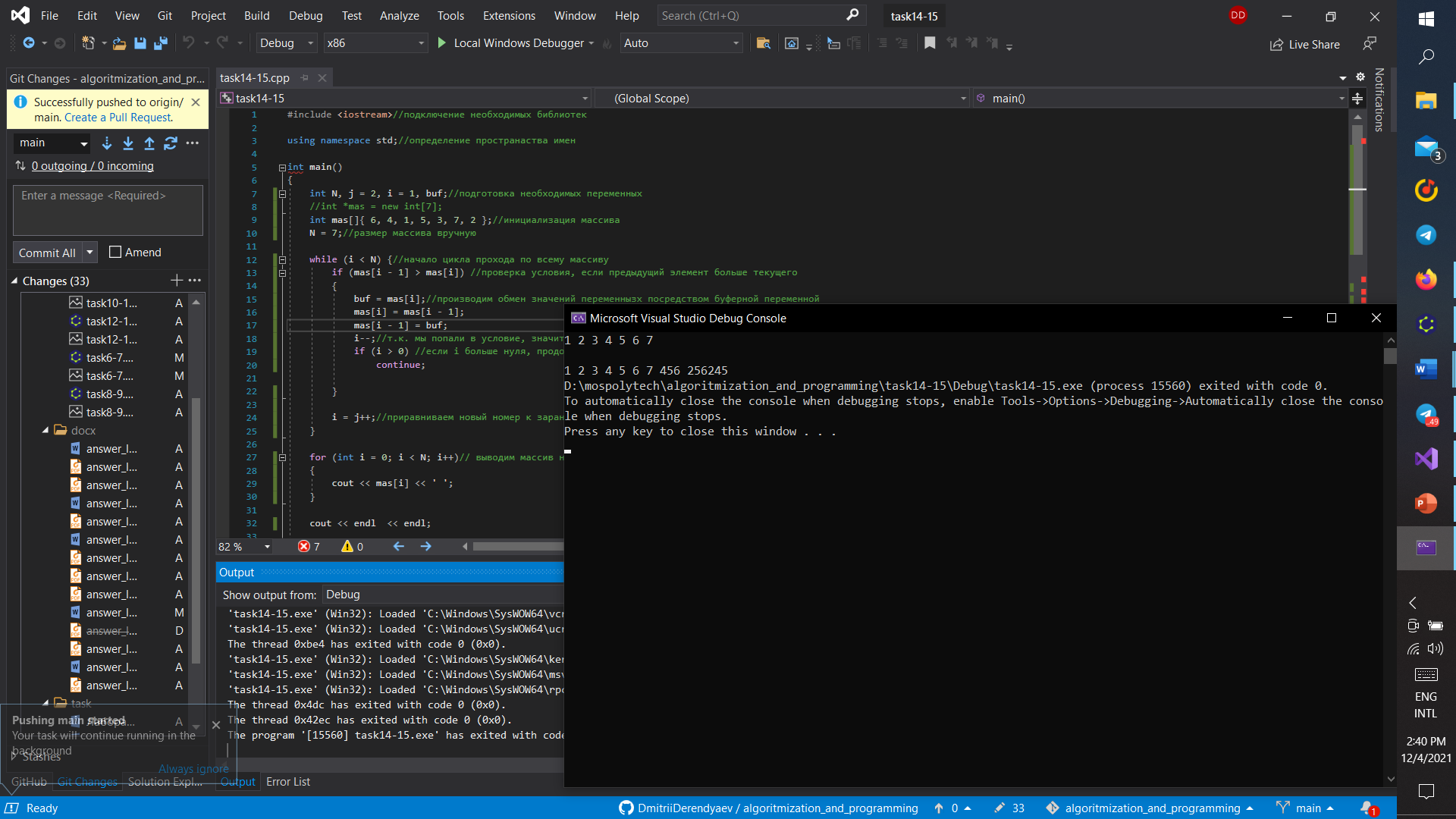
{

cout << mas2[i] << ' ';

}

return 0;

}



*При необходимости, вы можете найти всю историю разработки программы на моем GitHub:*

[*https://github.com/DmitriiDerendyaev/algoritmization\_and\_programming*](https://github.com/DmitriiDerendyaev/algoritmization_and_programming)